

## AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application:

### Listing of Claims:

1           1.       (Currently Amended) A method for reducing ~~the overhead-an~~  
2       ~~overhead~~ involved in executing native code methods in an application running on  
3       a virtual machine, comprising:  
4               selecting a call to any native code method to be optimized within the  
5       virtual machine;  
6               decompiling at least part of the ~~selected~~-native code method for the  
7       selected call into an intermediate representation, wherein an intermediate  
8       representation includes a set of instruction code which is not in final executable  
9       form;  
10              obtaining an intermediate representation associated with the application  
11       running on the virtual machine which interacts with the ~~selected~~-native code  
12       method for the selected call;  
13              integrating the intermediate representation for the ~~selected~~-native code  
14       method for the selected call into the intermediate representation associated with  
15       the application running on the virtual machine to form an integrated intermediate  
16       representation; and  
17              generating native-a native code from the integrated intermediate  
18       representation, wherein ~~the native code generation process optimizes generating~~  
19       the native code from the integrated intermediate representation involves  
20       optimizing interactions between the application running on the virtual machine  
21       and the ~~selected~~-native code method for the selected call, wherein optimizing the

22 interactions involves optimizing calls from the application to the ~~selected-native~~  
23 code method for the selected call by using additional information from the  
24 integrated intermediate representation to reduce ~~the number-a number~~ of indirect  
25 calls and indirect references associated with the calls from the application to the  
26 ~~selected-native~~ code method for the selected call.

1           2.       (Currently Amended) The method of claim 1, wherein selecting  
2 the call to any native code method involves selecting the call based upon at least  
3 one of:  
4           ~~the execution-an execution~~ frequency of the call; and  
5           ~~the overhead-an overhead~~ involved in performing the call ~~to the selected~~  
6 ~~native code method~~ as compared against ~~the amount-an amount~~ of work  
7 performed by the ~~selected-native~~ code method for the call.

1           3       (Canceled).

1           4.       (Currently Amended) The method of claim 1, wherein optimizing  
2 interactions between the application running on the virtual machine and the  
3 ~~selected-native~~ code method for the selected call involves optimizing callbacks by  
4 the ~~selected-native~~ code method for the selected call into the virtual machine.

1           5.       (Currently Amended) The method of claim 4, wherein optimizing  
2 callbacks by the ~~selected-native~~ code method for the selected call into the virtual  
3 machine involves optimizing callbacks that access heap objects within the virtual  
4 machine.

1           6.       (Currently Amended) The method of claim 4, wherein the virtual  
2 machine is a platform-independent virtual machine; and

1            wherein integrating the intermediate representation for the ~~selected-native~~  
2    code method for the selected call with the intermediate representation associated  
3    with the application running on the virtual machine involves integrating calls  
4    provided by an interface for accessing native code into the ~~selected-native code~~  
5    method for the selected call.

1            7.        (Original) The method of claim 1, wherein obtaining the  
2    intermediate representation associated with the application running on the virtual  
3    machine involves recompiling a corresponding portion of the application.

1            8.        (Original) The method of claim 1, wherein obtaining the  
2    intermediate representation associated the application running on the virtual  
3    machine involves accessing a previously generated intermediate representation  
4    associated with the application running on the virtual machine.

1            9.        (Currently Amended) The method of claim 1, wherein prior to  
2    decompiling the ~~selected-native code method for the selected call~~, the method  
3    further comprises setting up a context for the decompilation by:  
4            determining a signature of the ~~call to the selected-native code method; and~~  
5    selected call; and  
6            determining a mapping from arguments of the selected call to  
7    corresponding locations in a native application binary interface (ABI).

1            10.       (Currently Amended) A computer-readable storage device storing  
2    instructions that when executed by a computer cause the computer to perform a  
3    method for reducing ~~the overhead-an overhead~~ involved in executing native code  
4    methods in an application running on a virtual machine, the method comprising:

5           selecting a call to any native code method to be optimized within the  
6   virtual machine;  
7           decompiling at least part of the ~~selected~~-native code method for the  
8   selected call into an intermediate representation, wherein an intermediate  
9   representation includes a set of instruction code which is not in final executable  
10   form;  
11          obtaining an intermediate representation associated with the application  
12   running on the virtual machine which interacts with the ~~selected~~-native code  
13   method for the selected call;  
14          integrating the intermediate representation for the ~~selected~~-native code  
15   method for the selected call into the intermediate representation associated with  
16   the application running on the virtual machine to form an integrated intermediate  
17   representation; and  
18          generating ~~native-a native~~ code from the integrated intermediate  
19   representation, wherein ~~the native code generation process optimizes generating~~  
20   the native code from the integrated intermediate representation involves  
21   optimizing interactions between the application running on the virtual machine  
22   and the ~~selected~~-native code method for the selected call, wherein optimizing the  
23   interactions involves optimizing calls from the application to the ~~selected~~-native  
24   code method for the selected call by using additional information from the  
25   integrated intermediate representation to reduce ~~the number-a number~~ of indirect  
26   calls and indirect references associated with the calls from the application to the  
27   ~~selected~~-native code method for the selected call.

1           11.   (Currently Amended) The computer-readable storage device of  
2   claim 10, wherein selecting the call to any native code method involves selecting  
3   the call based upon at least one of:  
4           ~~the execution-an execution~~ frequency of the call; and

5           ~~the overhead~~an overhead involved in performing the call ~~to the selected~~  
6   ~~native code method~~ as compared against ~~the amount~~an amount of work  
7   performed by the ~~selected~~-native code method for the call.

1           12     (Canceled).

1           13.     (Currently Amended) The computer-readable storage device of  
2   claim 10, wherein optimizing interactions between the application running on the  
3   virtual machine and the ~~selected~~-native code method for the selected call involves  
4   optimizing callbacks by the ~~selected~~-native code method for the selected call into  
5   the virtual machine.

1           14.     (Currently Amended) The computer-readable storage device of  
2   claim 13, wherein optimizing callbacks by the ~~selected~~-native code method for the  
3   selected call into the virtual machine involves optimizing callbacks that access  
4   heap objects within the virtual machine.

1           15.     (Currently Amended) The computer-readable storage device of  
2   claim 13,  
3         wherein the virtual machine is a platform-independent virtual machine;  
4   and  
5         wherein ~~integrating~~integrating the intermediate representation for the  
6   ~~selected~~-native code method for the selected call with the intermediate  
7   representation associated with the application running on the virtual machine  
8   involves integrating calls provided by an interface for accessing native code into  
9   the ~~selected~~-native code method for the selected call.

1           16.     (Previously presented) The computer-readable storage device of  
2 claim 10, wherein obtaining the intermediate representation associated with the  
3 application running on the virtual machine involves recompiling a corresponding  
4 portion of the application.

1           17.     (Previously presented) The computer-readable storage device of  
2 claim 10, wherein obtaining the intermediate representation associated with the  
3 application running on the virtual machine involves accessing a previously  
4 generated intermediate representation associated with the application running on  
5 the virtual machine.

1           18.     (Currently Amended) The computer-readable storage device of  
2 claim 10, wherein prior to decompiling the ~~selected~~ native code method for the  
3 selected call, the method further comprises setting up a context for the  
4 decompilation by:  
5           determining a signature of the ~~call to the selected native code method;~~  
6 ~~and selected call; and~~  
7           determining a mapping from arguments of the selected call to  
8 corresponding locations in a native application binary interface (ABI).

1           19-27. (Cancelled)

1           28.     (Currently Amended) A method for reducing ~~the overhead~~ an  
2 overhead involved in executing native code methods in an application running on  
3 a virtual machine, comprising:  
4           deciding to optimize a callback by any native code method into the virtual  
5 machine;

6           decompiling at least part of the ~~selected~~-native code method for the  
7       callback into an intermediate representation, wherein an intermediate  
8       representation includes a set of instruction code which is not in final executable  
9       form;  
10          obtaining an intermediate representation associated with the application  
11       running on the virtual machine which interacts with the ~~selected~~-native code  
12       method for the callback;  
13          integrating the intermediate representation for the ~~selected~~-native code  
14       method for the callback into the intermediate representation associated with the  
15       application running on the virtual machine to form an integrated intermediate  
16       representation; and  
17          generating native-a native code from the integrated intermediate  
18       representation, wherein ~~the native code generation process optimizes the callback~~  
19       ~~by any native code method into the virtual machine~~, generating the native code  
20       from the integrated intermediate representation involves optimizing the callback,  
21       wherein optimizing the callback involves optimizing calls from the ~~selected~~  
22       native code method for the callback to the application by using additional  
23       information from the integrated intermediate representation to reduce ~~the number~~  
24       a number of indirect calls and indirect references associated with the calls from  
25       the ~~selected~~-native code method for the callback to the application.

1           29.     (Currently Amended) The method of claim 28, wherein ~~the native~~  
2       ~~code generation process also optimizes calls to the selected native code method~~  
3       ~~by the application~~, generating the native code from the integrated intermediate  
4       representation also involves optimizing calls by the application to the native code  
5       method for the callback.

1           30.     (Previously Presented) The method of claim 28, wherein  
2     optimizing the callback by any native code method into the virtual machine  
3     involves optimizing a callback that accesses a heap object within the virtual  
4     machine.

1           31.     (Currently Amended) The method of claim 28,  
2     wherein the virtual machine is a platform-independent virtual machine;  
3     and  
4     wherein integrating the intermediate representation for the ~~selected-native~~  
5     code method for the callback with the intermediate representation associated with  
6     the application running on the virtual machine involves integrating calls provided  
7     by an interface for accessing native code into the ~~selected-native code method~~ for  
8     the callback.

1           32.     (Currently amended) A computer-readable storage device storing  
2     instructions that when executed by a computer cause the computer to perform a  
3     method for reducing ~~the overhead-an overhead~~ involved in executing native code  
4     methods in an application running on a virtual machine, the method comprising:  
5         deciding to optimize a callback by any native code method into the virtual  
6     machine;  
7         decompiling at least part of the ~~selected-native code method~~ for the  
8     callback into an intermediate representation, wherein an intermediate  
9     representation includes a set of instruction code which is not in final executable  
10    form;  
11         obtaining an intermediate representation associated with the application  
12    running on the virtual machine which interacts with the ~~selected-native code~~  
13    method for the callback;



14 integrating the intermediate representation for the ~~selected~~ native code  
15 method for the callback into the intermediate representation associated with the  
16 application running on the virtual machine to form an integrated intermediate  
17 representation; and  
18 generating ~~native~~ a native code from the ~~combined-integrated~~ intermediate  
19 representation, wherein ~~the native code generation process optimizes the callback~~  
20 ~~by any native code method into the virtual machine,~~ generating the native code  
21 from the integrated intermediate representation involves optimizing the callback,  
22 wherein optimizing the callback involves optimizing calls from the ~~selected~~  
23 native code method for the callback to the application by using additional  
24 information from the integrated intermediate representation to reduce ~~the number~~  
25 a number of indirect calls and indirect references associated with the calls from  
26 the ~~selected~~ native code method for the callback to the application.

1 33. (Currently Amended) The computer-readable storage device of  
2 claim 32, wherein ~~the native code generation process also optimizes calls to the~~  
3 ~~selected native code method by the application,~~ generating the native code from  
4 the integrated intermediate representation also involves optimizing calls by the  
5 application to the native code method for the callback.

1 34. (Previously Presented) The computer-readable storage device of  
2 claim 32, wherein optimizing the callback by any native code method into the  
3 virtual machine involves optimizing a callback that accesses a heap object within  
4 the virtual machine.

1 35. (Currently Amended) The computer-readable storage device of  
2 claim 32, wherein the virtual machine is a platform-independent virtual machine;  
3 and

4            wherein integrating the intermediate representation for the ~~selected-native~~  
5   code method for the callback with the intermediate representation associated with  
6   the application running on the virtual machine involves integrating calls provided  
7   by an interface for accessing native code into the ~~selected-native~~ code method for  
8   the callback.

1            36-39. (Cancelled)